Extreme Computing
Assignment

1 Introduction

This is the coursework assignment for the Extreme Computing course 2019/20. You need to use the Hadoop Streaming API to solve problems you might encounter when working in a big data context. This section will give you administrative information and help with solving the assignment. This is followed by the actual tasks and finally a description of how to submit your solutions.

1.1 Administrative Information

Deadline The assignment is due **on Monday 21 October at 16:00**.

Deadline Extension The School of Informatics has a policy on coursework deadlines, which applies across all taught courses. Further information can be found here:

`http://web.inf.ed.ac.uk/infweb/student-services/ito/admin/coursework-projects/late-coursework-extension-requests`

Questions All questions should go on Piazza

`https://piazza.com/ed.ac.uk/fall2019/infr11088`

in one of the "assignment" folders. Feel free to discuss general techniques amongst each other unless you would reveal an answer. If your question / discussion reveals an answer, ask privately.

Assignment office hours We will provide office hours in week 2 and 3 after assignment release to answer questions in person regarding the assignment. Please look at the course's LEARN page for times and locations.

Marking The assignment is worth 100 marks in total and counts for 40% of the final course mark. Marks are given for correctness, efficiency and proper use of tools. This includes running your solution in a scalable way.

For purposes of correctness marks, we will normalise your output by concatenating files, stripping leading and trailing whitespace, and sorting. That means you can have a different partitioner, different sorting comparison function, and different number of reducers while getting correctness marks. However, if a question requires a certain order of elements as output or has only singular data output, you should use a single reducer in your last MapReduce job, i.a. a single output file. The same is true if the task specifically asks for such behaviour. We will also expect your code to behave correctly on other inputs.

Marks are indicated on the right margin of the page by two numbers, e.g. **10+15 marks**. The first number indicates achievable marks for correctness while the second number indicates achievable marks for efficiency.

Submission The submission process for your solution is described in Section 4. We start marking at the deadline and only mark once. If you are submitting on time, you can submit as many times as you want and the last one will be marked. If you are submitting late, you may only submit once in total (which implies that you should not submit before the deadline) or run the risk that we will mark an old version then penalise for the last submission time.

Marking Feedback You will receive your marks and feedback for your solution to your student email address within three weeks following your submission. Once you have received your marks, you have three days to ask questions about them, e.g. feedback clarification. Please post any such questions in a private Piazza message.

Good Scholarly Practice Please remember the University requirement regarding all assessed work for credit. Details about this can be found at:

`http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct`

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (generally permitting access only to yourself, or your group in the case of group practicals).

1.2 Teaching Hadoop Cluster

You should use the teaching Hadoop Cluster[1] and the Hadoop Streaming API with **Python 2.7** code for Mappers, Reducers and Combiners for all of your solutions. You will have local filespace (under your own credentials) in HDFS at `/user/UUN`

If you are logging in from outside Informatics, first

```
ssh s12345678@student.ssh.inf.ed.ac.uk
```

or use the Informatics VPN:

`http://computing.help.inf.ed.ac.uk/openvpn`

Once inside Informatics, connect to cluster's resource manager node `scutter02` to submit your jobs:

```
ssh hadoop.exc
```

To manage cluster jobs use the `mapred` command or view the web interface of the job tracker at:

`http://hadoop.exc.inf.ed.ac.uk:8088/cluster`

1.3 Official Resources and Supplementary Material

You are expected to find your own way to solve the tasks. You can use the internet to find resources which might help you, e.g. Hadoop tutorials or documentation. We will, however, provide supplementary material tailored to using the teaching Hadoop Cluster to get you started. You can find it on the course's `Learn` page under `Assessment` → `Assignment Information` → `Supplementary Material`

[1]You can run your own, but we won't support it and you need to copy output back to the teaching one.

1.4 General Hints

Task Runtime All of the solutions take less than an hour and less than 30 total workers (mappers + reducers). You should kill any jobs that take longer:

```
mapred job -kill $jobid
```

We may use a bot to kill non-conforming jobs.

Complexity If not otherwise stated, we generally expect O(1) space complexity for full marks in mappers, combiners and reducers.

Map-Reduce-Jobs If not otherwise stated, we expect a single Map-Reduce-Job for full marks. If multiple jobs are required, results from the first job can be used as input for a following job.

Expected Outputs Each question defines the expected output format by specifying expected fields and data types and whether singular or multiple outputs are expected. For multiple outputs, the types are enclosed in []. The following example expects multiple lines of two fields with integer and string value.

[value1:int|value2:str]

Output Separator The field separator for all output should be the UNIX pipe character '|'.

2 Dataset

For the dataset you will use in this assignment, there is a `large` repository to be used when deploying your solutions using Hadoop, and a `small` repository which you can copy to your local machine for local testing and debugging. All datasets can be found on the Teaching Hadoop Cluster in the `hdfs` file system under `/data`.

2.1 Internet Movie Database (IMDB)

This assignment will be on processing the IMDB dataset – we have chosen a subset for these tasks to encourage you to think about how to structure your solutions to use multiple input data streams, and efficiently process structured text using Hadoop Streaming.

Please note that we have removed the first line of each `tsv` file, which contained the column names in the original dataset. We have done this for your convenience, as in your code, you can assume all lines are data. The four files used, including their structures, are detailed in Section 2.2.

- `/data/small/imdb/*.tsv` – for development and local testing
- `/data/large/imdb/*.tsv` – for the final results
- `/data/samples/imdb` – sample results on the `small` data

Note that not all `.tsv` files are required for all questions. Consult the schema in Section 2.2 to ascertain which one(s) you require for the task at hand. Be aware that `skipVal` (`'\N'`) may be present where fields are denoted `Optional`, meaning no data is present. You are expected to account for this possibility and ignore those entries from your solutions.

We have alse provided the results from the solution of each task on the small data set for your convenience. However, note that a candidate solution that produces the same results as the samples on the small data set does not necessarily mean that it is complete (i.e. correct and efficient) for the large data set.

2.2 IMDB Schema Reference

The following table defines the columns in each of the provided files from the IMDB dataset to aid you in your solution design.

- `Optional[T]` means either type `T` is present, or `skipVal` (‘`\N`’) otherwise
- `List[T]` means a comma-delimited list of type `T` is present, e.g. ‘`dog,cat,bear`’, where `T := str`

INDEX	FIELD	TYPE	EXAMPLES/NOTES
			name.basics.tsv
0	`nconst`	`str`	`nmXXXXXXX` – Unique person/crew ID
1	`primaryName`	`Optional[str]`	–
2	`birthYear`	`Optional[int]`	–
3	`deathYear`	`Optional[int]`	–
4	`primaryProfession`	`Optional[List[str]]`	‘`editor,manager`’, ‘`actor`’, ‘`actress`’
5	`knownForTitles`	`Optional[List[str]]`	‘`tconst1,tconst2,tconst3`’
			title.basics.tsv
0	`tconst`	`str`	`ttXXXXXXX` – Unique title ID
1	`titleType`	`Optional[str]`	‘`tvMovie`’, ‘`short`’, ‘`movie`’, ‘`videoGame`’
2	`primaryTitle`	`Optional[str]`	–
3	`originalTitle`	`Optional[str]`	–
4	`isAdult`	`int`	–
5	`startYear`	`Optional[int]`	`YYYY` – Release year
6	`endYear`	`Optional[int]`	`YYYY` – End year, e.g. when a play ends.
7	`runtimeMinutes`	`Optional[int]`	–
8	`genres`	`Optional[List[str]]`	‘`Documentary,Short,Sport`’
			title.crew.tsv
0	`tconst`	`str`	Joins `title.basics.tconst`
1	`directors`	`Optional[List[str]]`	‘`nmXXXXXX1,nmXXXXXX2`’ – Joins `nconst`
2	`writers`	`Optional[List[str]]`	‘`nmXXXXXX1,nmXXXXXX2`’ – Joins `nconst`
			title.ratings.tsv
0	`tconst`	`str`	Joins `title.basics.tconst`
1	`averageRating`	`float`	–
2	`numVotes`	`int`	–

3 Tasks

Consult the schema in Section 2.2 when designing your solutions in order to extract the correct data.

Task 1

◁ **Task**

Calculate the average maximum and minimum runtime duration for all titles per movie genre.

Note that a title can have more than one genre, thus it should be considered for all of them. The results should be kept in minutes and the average value should be rounded to 2 decimal places (e.g. 25.30). Lastly, titles with 0 runtime duration are valid and should be accounted for in your solution. **(10+15 marks)**

[avg_runtime:float|max_runtime:int|min_runtime:int|genre:str]

Task 2

◁ **Task**

Print the titles of the movies which were released between 1990 and 2018 (inclusive), have an average rating of 7.5 or more, and have received 500000 votes or more.

For the titles use the `primaryTitle` field and account only for entries whose `titleType` is `‘movie’`. **(10+15 marks)**

[title:str]

Task 3

◁ **Task**

Print the top rated movie of each genre for each decade between 1900 and 1999.

For the titles use the `primaryTitle` field and account only for entries whose `titleType` is `‘movie’`. For calculating the top rated movies use the `averageRating` field and for the release year use the `startYear` field.

The output should be sorted by decade and then by genre. For the movies with the same rating and of the same decade, print only the one with the title that comes first alphabetically. Each decade should be represented with a single digit, starting with 0 corresponding to 1900-1909.

This task may require more than one MapReduce job. For full marks, you should not need more than two. **(10+15 marks)**

[decade:int|genre:str|title:str]

Task 4

◁ **Task**

Print the names of the top 10 most popular writers, based on the votes of their single most popular title they are known-for as writers.

For example, John known-for (as a writer) of a single title that received ten votes ranks higher than Bob who is known-for (as a writer) for two titles with eight votes each.

Note that a writer who is known for certain titles (see the `knownForTitles` field) does not necessarily mean that they contributed in that capacity in the creation of each of those titles. Thus, apart from a person being known for a title, they should also be among the writers of that title (see `title.crew.tsv` file).

The output should contain 10 lines in descending order based on the number of votes of individual titles, as indicated by the example. No duplicate writer names are allowed.

This task may require more than one MapReduce job. For full marks, you should not need more than two. **(10+15 marks)**

[votes:int|writer_name:str]

4 Submissions

To submit your work, please do the following:

1. Make sure that you store the output of your MAPREDUCE jobs in HDFS. Please store the output for the Xth task in the folder: `/user/sXXXXXXX/assignment/taskX` (replace `sXXXXXXX` with your student number). This way we can easily check whether you got the right output. When marking, we will only be interested in the output for the **large** version inputs. Do not put the output for the small version inputs in the same folder as it can be mistaken for your output for the large version. **Do not delete these files from HDFS until you are told it is OK to do so.**

2. To submit your code, please prepare a directory for submission that will contain one directory for each task and name the directory for the Xth task `taskX`. Inside each task folder, please include:

   - One file with `.sh` extension. This should be the shell script that executes your MAPREDUCE job(s) for Xth task.
   - One file with `.out` extension. This file should contain the first 20 lines of your output. If Hadoop splits your output into multiple files, please use the first twenty lines from the first output file (usually called `part-00000`). You can use the command:

     ```
     hdfs dfs -cat filename | head -20
     ```

     for showing just the first 20 lines of the file. If the whole output has less then 20 lines together, then your `*.out` file should contain the whole output.
   - Files with your code. This includes all code that you wrote to solve the task. **Do not submit a Word document or a PDF file—only submit a plain text file**. It is not required, but it will make marking easier if the file name with mapper, combiner, reducer code starts with `mapper`, `combiner` and `reducer` respectively.
     You can see an example below:

     ```
     submissionFolder/
     -- task1/
        -- run.sh
        -- output.out
        -- mapper.py
        -- mapper2.py
        -- reducer.py
        -- reducer2.py
        -- combiner.py
        -- combiner2.py
     -- task2/
     ...
     ```

     This is an artificial example, it does not mean that you need two MAPREDUCE jobs with combiners to solve the first task.

   Once you have prepared your submission directory with the specified structure, please run

   ```
   /afs/inf.ed.ac.uk/group/teaching/exc/submit-assignment.sh path/to/submission_folder
   ```

   This will give you friendly warnings to help you spot if a folder (e.g. `task8`) or a file is missing. You will be asked whether you want to submit your submission directory (regardless of whether any warnings were given).

   NOTE: If you submit using `ssh` to access to the university network, make sure you are on `student.compute`. The gateway server does not have all the required tools for the script to work.